**Module 2 – Frontend HTML**

**HTML Basics**

**Theory Assignment**

**Question 1: Define HTML. What is the purpose of HTML in web development?**

**Definition of HTML:**

**HTML (Hyper Text Markup Language)** is the standard markup language used to create and structure content on the web. It uses a system of **tags and elements** to define text, images, links, and other content types, allowing web browsers to display them properly.

**Purpose of HTML in Web Development:**

1. **Defines Structure**: HTML gives webpages their structure using elements like <head>, <body>, <div>, <h1> to <h6>, <p>, etc.
2. **Displays Content**: It allows developers to add and organize content such as text, images, links, and media.
3. **Supports Navigation**: HTML enables linking between pages and sections using anchor (<a>) tags.
4. **Improves Accessibility & SEO**: Semantic HTML helps screen readers and search engines understand the content better.
5. **Foundation for Web Technologies**: It works alongside CSS (for styling) and JavaScript (for interactivity) to build complete, functional websites.

**Question 2: Explain the basic structure of an HTML document. Identify the mandatory tagsand their purposes.**

The basic structure of an **HTML (HyperText Markup Language)** document includes a set of essential tags that define the content and structure of a web page. Here’s an explanation of the basic structure and the **mandatory tags**:

**Basic Structure of an HTML Document**

<!DOCTYPE html>

<html>

<head>

<title>Page Title</title>

</head>

<body>

<h1>Main Heading</h1>

<p>This is a paragraph.</p>

</body>

</html>

**Mandatory Tags and Their Purposes**

1. **<!DOCTYPE html>**
   * Declares the document type and HTML version.
   * Tells the browser to render the page in standards mode using HTML5.
2. **<html>**
   * Root element of the HTML document.
   * Encloses all other elements (except <!DOCTYPE>).
3. **<head>**
   * Contains metadata about the document (not displayed directly on the web page).
   * Includes elements like <title>, <meta>, <link>, <style>, and <script>.
4. **<title>**
   * Sets the title of the page that appears in the browser tab.
   * Must be placed within the <head> tag.
5. **<body>**
   * Contains the visible content of the web page, such as text, images, links, and other elements.

**Question 3: What is the difference between block-level elements and inline elements in HTML? Provide examples of each.**

**1. Block-Level Elements**

**Definition:**  
Block-level elements start on a **new line** and take up the **full width** available (by default), stretching from the left to the right edge of their container.

**Key Characteristics:**

* Always start on a new line.
* Can contain other block-level or inline elements.
* Commonly used to define structural parts of a page.

**Examples:**

* <div> – generic container
* <p> – paragraph
* <h1> to <h6> – headings
* <ul>, <ol> – list’s
* <li> – list item
* <section>, <article>, <nav>, <footer> – semantic structural elements

<div>This is a block-level element.</div>

<ul>

<li>Home</li>

<li>About</li>

<li>Service</li>

</ul>

<h1>head</h1> to <h6>head</h6>

**2. Inline Elements**

**Definition:**  
Inline elements do **not start on a new line**. They only take up as much **width** as necessary, flowing **within** a line of text.

**Key Characteristics:**

* Do not break the flow of content.
* Usually used for formatting small chunks of content.
* Cannot contain block-level elements (only text or other inline elements).

**Examples:**

* <span> – generic inline container
* <a> – anchor (link)
* <strong>, <em> – bold and italic emphasis
* <img> – image
* <br> – line break

**Example Code:**

<p>This is a <strong>bold</strong> word inside a paragraph

<a href=”#”>inline link</a>

</p><br>

**Question 4: Discuss the role of semantic HTML. Why is it important for accessibility and SEO? Provide examples of semantic elements**.

**Semantic HTML** refers to the use of HTML elements that clearly describe their meaning and purpose within a web page. Unlike non-semantic elements like <div> and <span>, semantic elements such as <article>, <nav>, and <footer> provide context and structure to the content they enclose.

**Why Semantic HTML Is Important**

**1. Accessibility**

Semantic HTML enhances the experience for users who rely on **assistive technologies** (like screen readers). These technologies interpret the structure and meaning of a page through semantic elements, helping users:

* Navigate pages more efficiently (e.g., jumping to the <main> or <nav> section)
* Understand the purpose of different content areas
* Identify headings, sections, and articles

This is crucial for users with visual impairments or cognitive disabilities.

**2. SEO (Search Engine Optimization)**

Search engines use semantic HTML to better **index and rank content**. When the structure of a webpage is clear:

* Search engines can understand which parts are important (e.g., <h1>, <article>, <section>)
* Content relevance is improved for keyword matching
* Snippets and previews in search results are more accurately generated

Well-structured semantic HTML contributes to better **search engine visibility and ranking**.

**Examples of Semantic HTML Elements**

| **Element** | **Purpose** |
| --- | --- |
| <header> | Represents the introductory content or a group of navigational links |
| <nav> | Contains navigation links |
| <main> | Indicates the main content area of a page |
| <section> | Represents a thematic grouping of content with a heading |
| <article> | Encapsulates independent, self-contained content |
| <aside> | Represents side content, like a sidebar or callout box |
| <footer> | Defines footer content for its nearest section or the entire page |
| <figure> | Groups visual content (like an image) with an optional caption |
| <figcaption> | Provides a caption or description for the <figure> element |
| <mark> | Highlights or emphasizes a portion of text |

**Examples of Semantic HTML**

<html>

<head>

<title>documents</title>

</head>

<article>

<header>

<h1>Blog Post Title</h1>

<p>Top’s Classes</p>

</header>

<section>

<p>This is the main content of the blog post </p>

</section>

<footer>

<p>Join on class may 4, 2025</p>

</footer>

</article>

**HTML Forms**

**Theory Assignment**

**Question 1: What are HTML forms used for? Describe the purpose of the input, textarea, select, and button elements.**

**HTML forms** are used to **collect user input** on a web page and submit that data to a server for processing. Forms are essential for enabling interaction between the user and a website, such as:

* Logging in
* Registering for an account
* Searching for content
* Providing feedback or comments
* Placing orders or making reservations

Forms are defined using the <form> element, which wraps around input elements and often includes an action (URL where data is sent) and a method (GET or POST).

**Purpose of Key Form Elements**

| **Element** | **Purpose** |
| --- | --- |
| <input> | A versatile form element used to collect various types of user data like text, email, passwords, dates, checkboxes, etc. Example: <input type="text">, <input type="email">, <input type="checkbox"> |
| <textarea> | Used for multi-line text input, such as comments, messages, or descriptions. Unlike <input>, which is single-line, <textarea> allows users to enter longer blocks of text. |
| <select> | Creates a drop-down list of options. Often used when the user needs to choose one (or more) predefined options. Inside <select>, multiple <option> tags define the choices. |
| <button> | Represents a clickable button. It can be used to submit a form, reset the form, or run custom JavaScript. Example: <button type="submit">Submit</button> |

**Example Usage in a Form**

<html>

<head>

<title>document</title>

</head>

<body>

<form action="/submit" method="post">

<label>Name: <input type="text" name="username"></label><br>

<label>Message: <textarea name="message"></textarea></label><br>

<label>Favorite Color:

<select name="color">

<option value="red">Red</option>

<option value="blue">Blue</option>

</select>

</label><br>

<button type="submit">Send</button>

</form>

</body>

</html>

**Question 2: Explain the difference between the GET and POST methods in form submission. When should each be used?**

When a user submits a form, the data is sent to the server using one of two main HTTP methods: **GET** or **POST**. These methods determine **how the data is sent**, and **when each should be used** depends on the purpose of the form.

**🔹 GET Method**

**Definition:**  
The **GET** method appends form data to the **URL** in name/value pairs (as a query string).

**Characteristics:**

* Data is visible in the URL (e.g., example.com/search?query=books)
* Limited data length (because of URL length restrictions)
* Can be bookmarked or shared
* Typically used for **retrieving data** (not changing anything on the server)

**When to Use:**

* Search forms
* Filter or sort options
* When data does **not** need to be private or secure

**Example:**

<html>

<head>

<title>document</title>

</head>

<body>

<form action="/search" method="get">

<input type="text" name="query">

<button type="submit">Search</button>

</form>

</body>

</html>

**🔹 POST Method**

**Definition:**  
The **POST** method sends form data in the **body** of the HTTP request, not in the URL.

**Characteristics:**

* Data is not visible in the URL
* No size limitations (suitable for large inputs like files or long text)
* More secure for sensitive data
* Used for **submitting** or **modifying** data on the server (e.g., creating a user, posting a comment)

**When to Use:**

* Login forms
* Registration forms
* Feedback or comment submission
* File uploads

**Example:**

<html>

<head>

<title>document</title>

</head>

<body>

<form action="/submit-form" method="post">

<input type="text" name="username">

<button type="submit">Submit</button>

</form>

</body>

</html>

**Question 3: What is the purpose of the label element in a form, and how does it improve accessibility?**

The <label> element in HTML forms is used to **associate a text description with a form control** (such as an <input>, <select>, or <textarea>). This helps users understand what information is being requested.

**🔹 Purpose of the <label> Element**

* Provides a **human-readable name** for a form field
* Helps users know what to enter into a form control
* Allows users to **click the label** to focus or activate the associated form control, improving usability

**🔹 How <label> Improves Accessibility**

The <label> element is especially important for users who rely on **screen readers** or other assistive technologies. Here's how it helps:

1. **Screen Readers**:  
   Labels are read aloud by screen readers, allowing visually impaired users to understand what the input field is for.
2. **Clickable Area**:  
   Clicking a label moves focus to the input field it is associated with, which is helpful for users with motor disabilities or on mobile devices.
3. **Improved Form Navigation**:  
   Assistive technology can better interpret form structure and make navigation smoother.

**🔹 Proper Usage of <label>**

There are two main ways to associate a label with a form control:

* 1. **Using the for Attribute (Explicit Association)**

<html>

<head>

<title>document</title>

</head>

<body>

<form>

E-mail:<input type="email" id="email" name="email">

</form>

</body>

</html>

* 1. **Wrapping the Form Control (Implicit Association)**

<html>

<head>

<title>document</title>

</head>

<body>

<form>

E-mail:<input type="email" name="email">

</form>

</body>

</html>

**HTML Tables**

**Theory Assignment**

# 

**Question 1: Explain the structure of an HTML table and the purpose of each of the following elements: <table>,<tr>,<th>,<td>and<thead>.**

An **HTML table** is used to display data in a structured grid of **rows** and **columns**. The table is made up of several elements that define its layout and meaning.

**🔹 Basic Structure of an HTML Table**

<html>

<head>

<title>document</title>

</head>

<body>

<table>

<thead>

<tr>

<th>Header 1</th>

<th>Header 2</th>

</tr>

</thead>

<tr>

<td>Row 1, Cell 1</td>

<td>Row 1, Cell 2</td>

</tr>

<tr>

<td>Row 2, Cell 1</td>

<td>Row 2, Cell 2</td>

</tr>

</table>

</body>

</html>

**🔹 Purpose of Key Elements**

| **Element** | **Purpose** |
| --- | --- |
| <table> | The **container** for all table elements. It defines the table structure and holds rows, headers, and data cells. |
| <tr> (Table Row) | Represents a **row** of cells within the table. It can contain <th> (header cells) or <td> (data cells). |
| <th> (Table Header) | Defines a **header cell**. Text inside a <th> is **bold** and **centered by default**. Used for column or row headings and helps with accessibility. |
| <td> (Table Data) | Represents a **standard data cell** in the table. It holds the actual content like text, numbers, or images. |
| <thead> | Groups the **header section** of the table. It improves structure and accessibility and can help browsers style or scroll tables more effectively. |

**Question 2: What is the difference between colspan and rowspan in tables? Provide examples.**

In HTML tables, the colspan and rowspan attributes are used to **merge table cells** across multiple **columns** or **rows**, respectively. This helps in creating more complex and visually organized table layouts.

**🔹 1. colspan (Column Span)**

* **Definition**: Merges **multiple columns** into a single cell.
* **Use Case**: When you want a cell to stretch **horizontally** across multiple columns.

**Example:**

<html>

<head>

<title>document</title>

</head>

<body>

<table border="1">

<tr>

<th colspan="2">Name</th>

<th>Age</th>

</tr>

<tr>

<td>First</td>

<td>Last</td>

<td>25</td>

</tr>

</table>

</body>

</html>

**Explanation**:  
In the first row, the "Name" cell spans across two columns (First and Last), forming a single wide header.

**🔹 2. rowspan (Row Span)**

* **Definition**: Merges **multiple rows** into a single cell.
* **Use Case**: When you want a cell to stretch **vertically** across multiple rows.

**Example:**

<html>

<head>

<title>document</title>

</head>

<body>

<table border="1">

<tr>

<th rowspan="2">Name</th>

<td>First: John</td>

</tr>

<tr>

<td>Last: Doe</td>

</tr>

</table>

</body>

</html>

**Explanation**:

The "Name" cell spans two rows, vertically aligning with both "First" and "Last" name cells.

**Question 3: Why should tables be used sparingly for layout purposes? What is a better alternative?**

Using **HTML tables for layout** (such as arranging page sections or positioning elements) is an outdated and discouraged practice. While it was common in the early days of web design, it's now considered **bad practice** for several important reasons.

**🔹 Problems with Using Tables for Layout**

1. **Poor Accessibility**
   * Tables are meant to represent tabular data. When misused for layout, screen readers and assistive technologies may interpret the layout as actual data, which **confuses users with disabilities**.
2. **Rigid and Hard to Maintain**
   * Table-based layouts are difficult to edit, especially when making changes to structure or design.
   * They're not flexible or responsive to different screen sizes (e.g., mobile devices).
3. **Slower Page Load and Performance**
   * Tables render more slowly in some browsers because the entire table needs to load before being displayed.
4. **Semantic Confusion**
   * Mixing data and layout in tables **violates semantic HTML principles**, making the code harder to understand and maintain.

**🔹 Better Alternative: CSS for Layout**

Modern web design uses **CSS (Cascading Style Sheets)** to control layout and positioning. CSS provides tools that are:

* **Responsive** (e.g., media queries, flexbox, grid)
* **Accessible**
* **Easier to style and maintain**

**Examples of CSS Layout Alternatives:**

| **CSS Technique** | **Purpose** | **Example Use Case** |
| --- | --- | --- |
| **Flexbox** | One-dimensional layout (row or column) | Navigation bars, alignment of items |
| **Grid Layout** | Two-dimensional layout (rows + columns) | Complex page structures, galleries |
| **Media Queries** | Responsive design for various screen sizes | Mobile-first design |

**Example (Flexbox):**

<html>

<head>

<title>document</title>

</head>

<body>

<table>

<div style="display: flex;">

<div>Left Panel</div>

<div>Right Panel</div>

</div>

</table>

</body>

</html>